JSP (Java Sever Page)

- JSP로 시작하는 웹 프로그래밍

01. JSP 개요

2. JSP 학습에 필요한 기술

- **JSP는 웹 프로그래밍 기술로 HTML, 자바스크립트, CSS와 같은 기본 웹 프로그래밍 경험이 요구됨.**

기술	설명	요구 수준
HTML	클라이언트 기술로서, 웹 프로그램의 기본이 되며 시각적인 부분을 담당한다.	• HTML 문서 구조와 기본 태그 • FORM 관련 태그 • HTML5 기본 구조
자바스크립트	웹 화면과 사용자와의 상호작용 및 동적 웹 페이지를 구현할 때 필요한 기술이다.	• 기본 문법 • 객체와 메서드 • 내장 객체 • 이벤트 핸들링
CSS	웹 화면의 레이아웃과 디자인 요소를 구현할 때 필요한 기술이다.	• 스타일시트 정의 및 셀렉터 이해 • DOM 연동에 의한 동적 스타일 제어

01. JSP 개요

- **JSP는 자바언어 기반이며 개발 시 순수 자바 코드가 50% 이상으로 탄탄한 자바 기본기가 요구됨.**

기술	설명	요구 수준	
자바	소스코드를 작성하기 위한 프로그래밍 기본 언어로서, Java SE를 기준으로 한다.	• 자바 기본 • 상속, 오버로딩, 오버라이딩 • java.util, java.io 패키지 • 예외 핸들링	• 객체지향 개념 • 인터페이스 구현 • 스레드
JDBC	Java DataBase Connectivity의 약자로서, 자바에서 데이터베이스 프로그래밍을 하기 위한 기술이다.	• JDBC 드라이버 세팅 • PreparedStatement • 기초 SQL문	• ResultSet • 데이터 핸들링
서블릿	JSP의 기본이 되는 자바 기반의 웹 프로그래밍 핵심 기술이다.	• 서블릿 구조 이해 • request, response 처리	• 간단한 서블릿 프로그래밍 • GET/POST 처리

01. JSP 개요

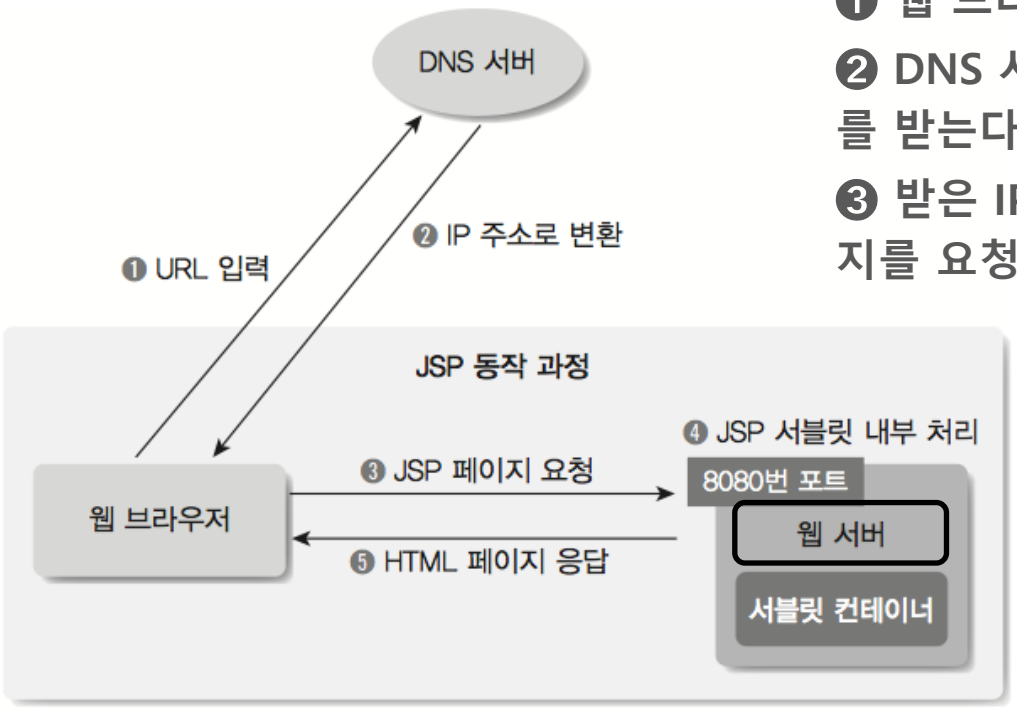
- **이외 추가적으로 다음 기술들에 대한 경험이 있다면 고급 웹 프로그래밍 학습에 도움이 됨.**

기술	필요성	요구 수준
데이터베이스	프로그램의 데이터를 처리하려고 할 때 반드시 필요하다.	• 다양한 SQL문의 사용 • 데이터베이스 연계 프로그래밍 경험 • 데이터베이스 함수 및 내장 프로시저
XML JSON	eXtensible Markup Language의 약자로서, 확장 가능한 구조적 문서 표현을 제공한다. 많은 프로그램에서 데이터 구조를 XML 기반으로 처리한다.	• XML 스키마 및 DTD 이해 • XML DOM 개요
모바일 프로그래밍	최근에는 스마트폰을 중심으로 하는 모바일 기반의 개발이 증가하고 있는 추세다.	• 안드로이드 혹은 아이폰 앱 개발 경험 • 하이브리드 앱 개발 경험
프레임워크	개발자로 하여금 더욱 좋은 프로그램을 만들 수 있도록 미리 제공되는 틀을 말한다.	• 소프트웨어 아키텍처 이해 • 스프링 프레임워크 • 스프링 @MVC

02. JSP 처리 과정의 이해

1. JSP 전체 동작 과정

- JSP 는 HTML 과 유사한 처리 과정을 거치나 HTML이 단순 서버 파일을 브라우저로 보내주는 것에 비해 JSP는 서버에서 프로그램이 실행된 결과를 웹 브라우저로 전달하는 차이가 있음.



❶ 웹 브라우저에서 URL을 입력한다.

❷ DNS 서버로부터 입력한 URL을 변환한 IP 주소를 받는다.

❸ 받은 IP 주소의 웹 서버 8080번 포트에 JSP 페이지를 요청한다.

❹ 웹 서버가 요청 내용을 분석하고 서블릿 컨테이너에 요청을 넘겨 처리한다.

❺ 화면에 보일 내용을 HTML 문서 형태로 웹 브라우저에 전송한다.

02. JSP 처리 과정의 이해

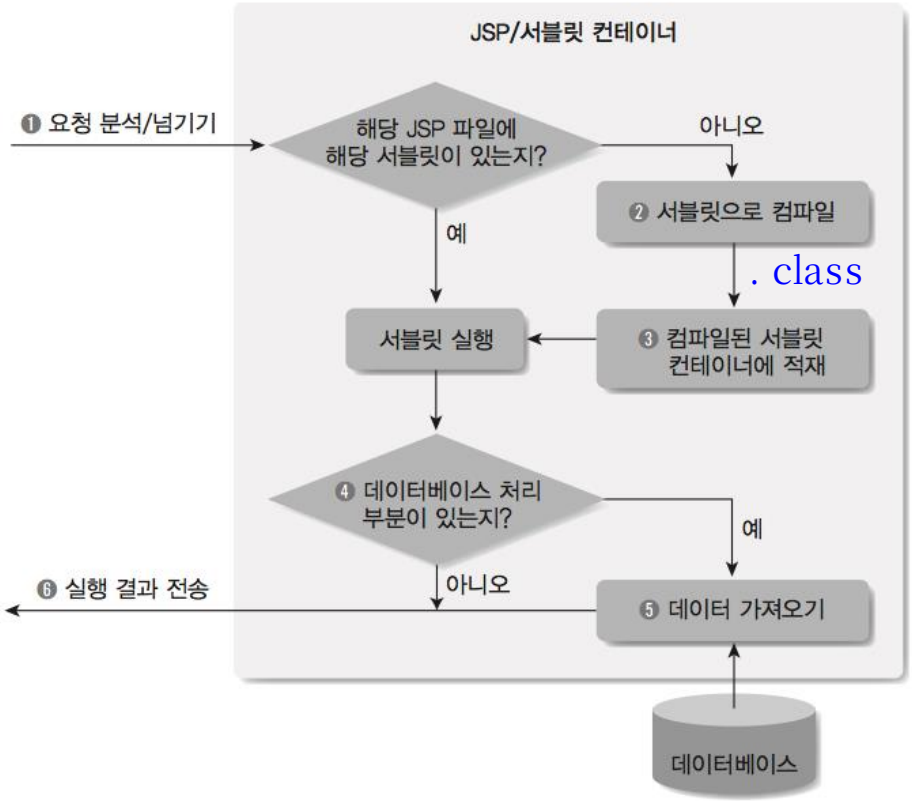
2. 서블릿 컨테이너 내부 과정

- JSP와 서블릿 차이
  - JSP는 HTML과 같은 일반적인 텍스트 파일 구조
  - 서블릿은 자바 소스로 작성된 클래스 파일 구조
  - **JSP는 서블릿 컨테이너에 의해 서블릿 형태의 자바 소스로 변환되어 클래스로 컴파일 됨**

서블린 컨테이너의 역할 .class - > 서블릿 객체 생성ㅇ
실행
소멸

- 서블릿 컨테이너
  - **서블릿 컨테이너는 서블릿을 실행하고 JSP를 서블릿 코드로 변환하는 기능을 수행함.**
  - 변환된 JSP의 서블릿 클래스를 실행하고 **웹 서버의 메모리에 적재하고 사용자 요청에 따라 실행.**

02. JSP 처리 과정의 이해



❶ 웹 서버로부터 JSP에 대한 사용자 요청이 컨테이너로 전달된다.

❷ 요청 JSP에 대한 서블릿이 존재하면 다음 단계로 진행하고, 존재하지 않을 경우 JSP를 .java 파일로 변환한 다음 .class 파일로 컴파일 한다.

❸ 컴파일된 서블릿 클래스를 컨테이너의 메모리에 적재하고 실행한다.

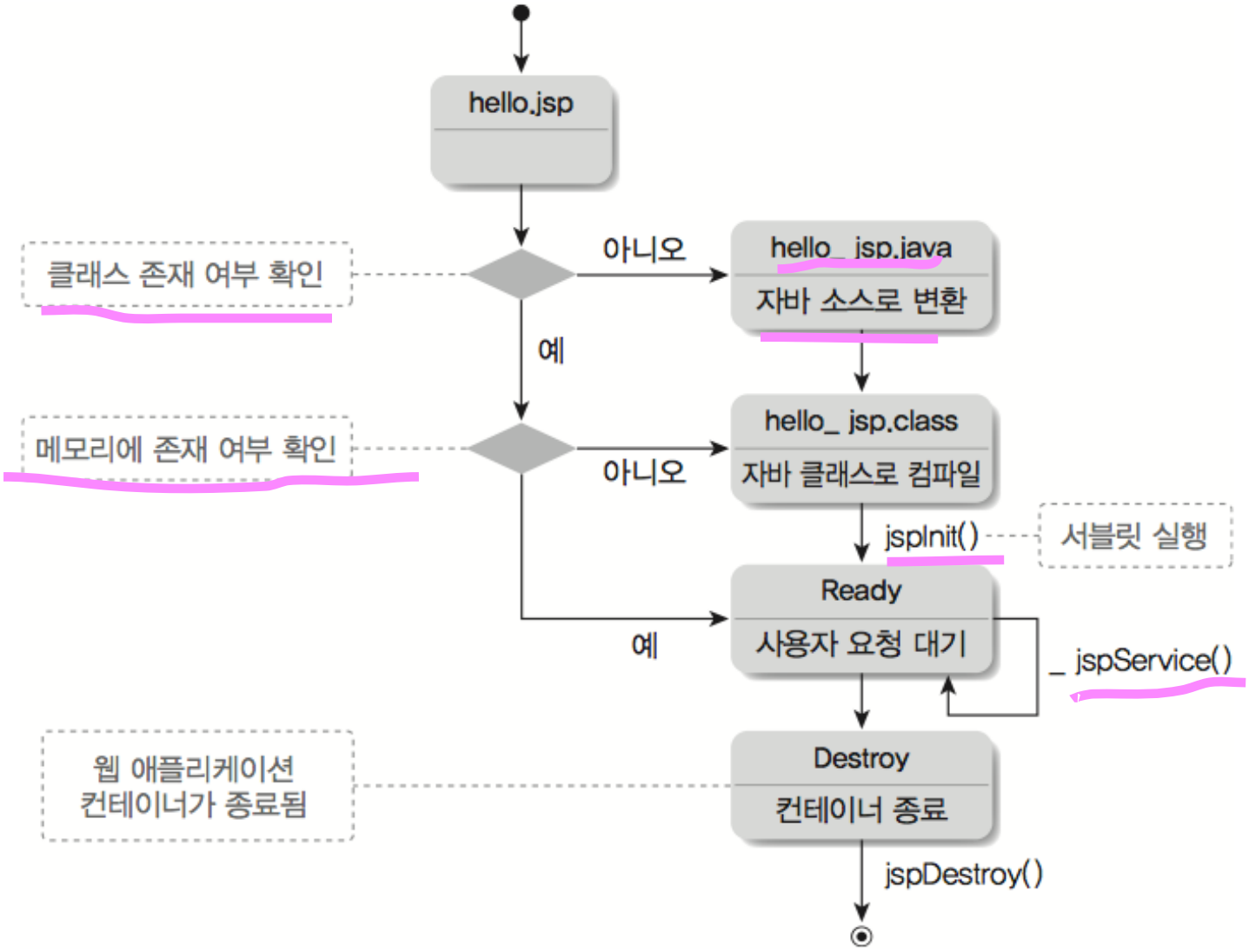
❹~❺ 데이터베이스 처리 혹은 별도의 기능을 위한 클래스 호출 등이 있다면 실행하고 결과를 취합해 HTML 형태로 구성한다.

❻ HTML 형태의 결과를 웹 서버를 경유해 사용자 브라우저에 전달한다.

02. JSP 처리 과정의 이해

- JSP 파일은 일반 텍스트를 비롯해 HTML 코드와 특수태그, 자바 코드가 섞여 있다.
- 서블릿으로 변환된 JSP는 컨테이너에 의해 생명주기가 관리된다.
- 서블릿으로 변환된 JSP는 jspInit() 메서드에 의해 실행되고 Ready 상태가 되며 이후 사용자 요청은 _jspService() 메서드가 쓰레드 형태로 호출되어 실행된다.
- 컨테이너에 의해 JSP 서블릿이 종료될 때에는 jspDestroy() 메서드가 실행된다.

02. JSP 처리 과정의 이해



02. JSP 처리 과정의 이해

- **JSP에 관해서 이것만은 알고 있자.**

자바 코드로 변경이된다

1. JSP는 일반 텍스트 파일로 되어 있다(텍스트 파일은 컴퓨터가 이해할 수 없다.
   즉 실행 가능한 프로그램이 아니며 특정 동작을 할 수 없다).

2. JSP는 HTML 코드와 몇몇 특수한 태그, 그리고 자바 코드가 섞여 있다.

3. 사용자가 요청할 경우 JSP는 컨테이너(톰캣)에 의해
   서블릿 형태의 .java 소스로 변환되고 컴파일된다.

4. 컴파일된 .class는 컴퓨터에서 실행할 수 있는 형태로 특정한 기능을 수행할 수 있게 된다.
   이후 소스 변경 전까지 해당 파일은 메모리에 상주하면서 다시 컴파일 되지 않고 서비스된다.

05. Servlet

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class HelloWorldServlet extends HttpServlet {
        public void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {

                response.setContentType("text/html; charset=EUC_KR");

                PrintWriter out = response.getWriter();
                out.println("<HTML><HEAD><TITLE>로그인</TITLE></HEAD>");
                out.println("<BODY><H2>Hello World : 헬로월드</H2>");
                out.println("오늘의 날짜와 시간은 : " + new java.util.Date());
                out.println("</BODY></HTML>");
        }
}
```

Apache.tomcat

05. Servlet

```
Wewcontent
        WEB-INF
                web.xml    ← 파일에 servlet 등록
```

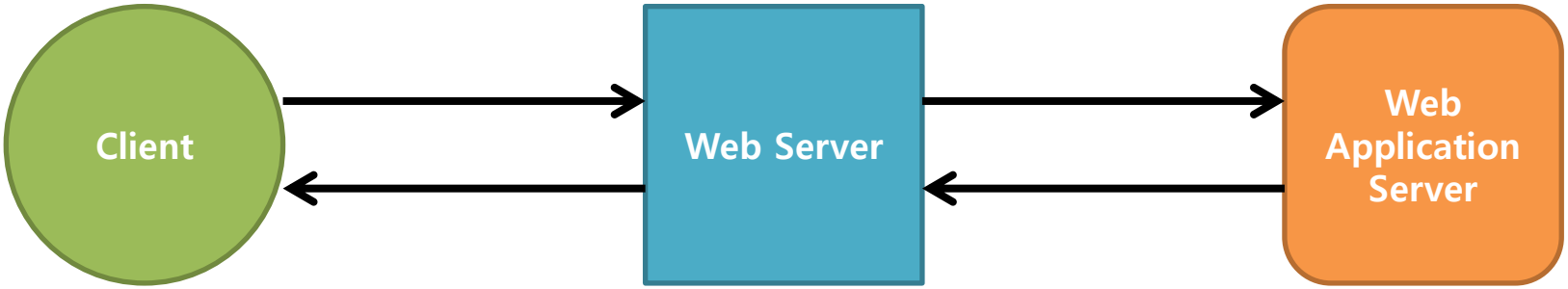
등록

```
<servlet>
        <servlet-name>NowServlet</servlet-name>
        <servlet-class>test.NowServlet</servlet-class>
</servlet>
```

url 매핑

```
<servlet-mapping>
        <servlet-name>NowServlet</servlet-name>
        <url-pattern>/now</url-pattern>
</servlet-mapping>
```

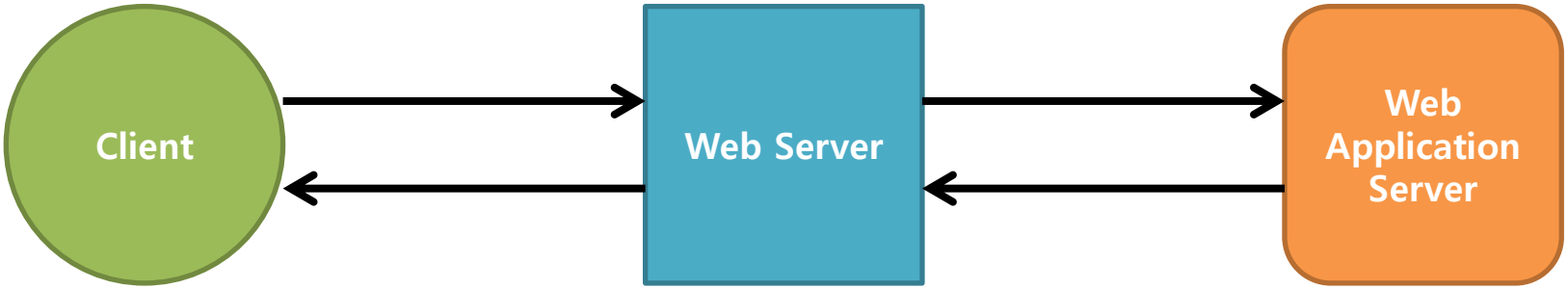
05. Servlet



05. JSP

```jsp
<%@ page contentType="text/html;charset=utf-8"%>
<HTML>
        <HEAD>
                <TITLE>Hello World</TITLE>
        </HEAD>
        <BODY>
                <H2>Hello World : 헬로월드</H2>
                오늘의 날짜와 시간은 :
                <%=new java.util.Date()%>
        </BODY>
</HTML>
```

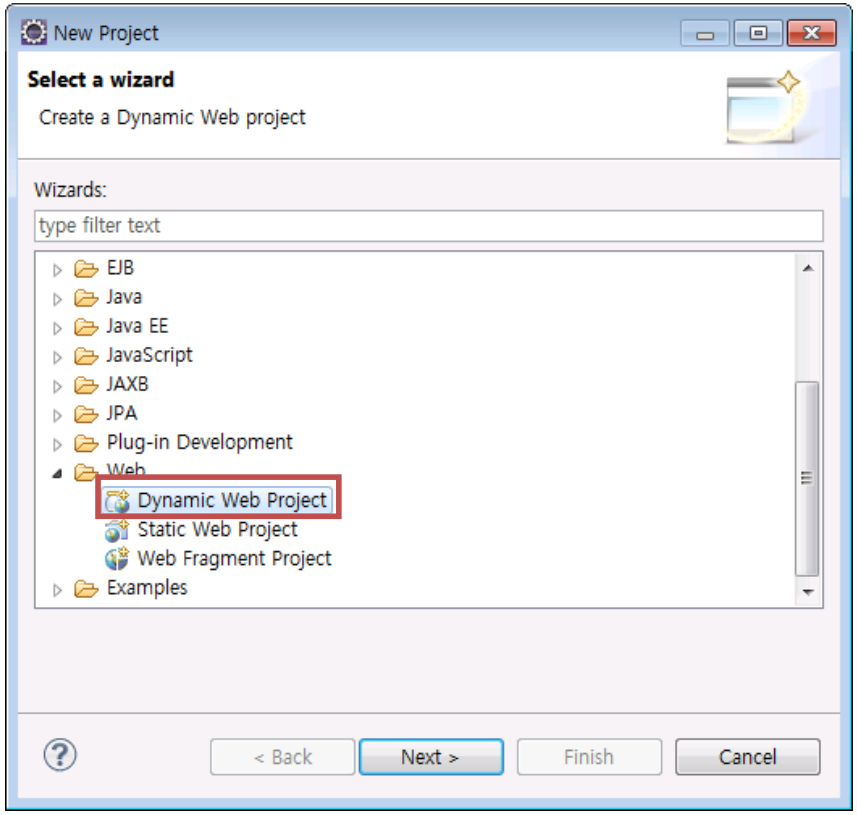
05. JSP



04. [기본실습]JSP 프로그래밍: Hello World JSP

1. 이클립스 프로젝트 생성

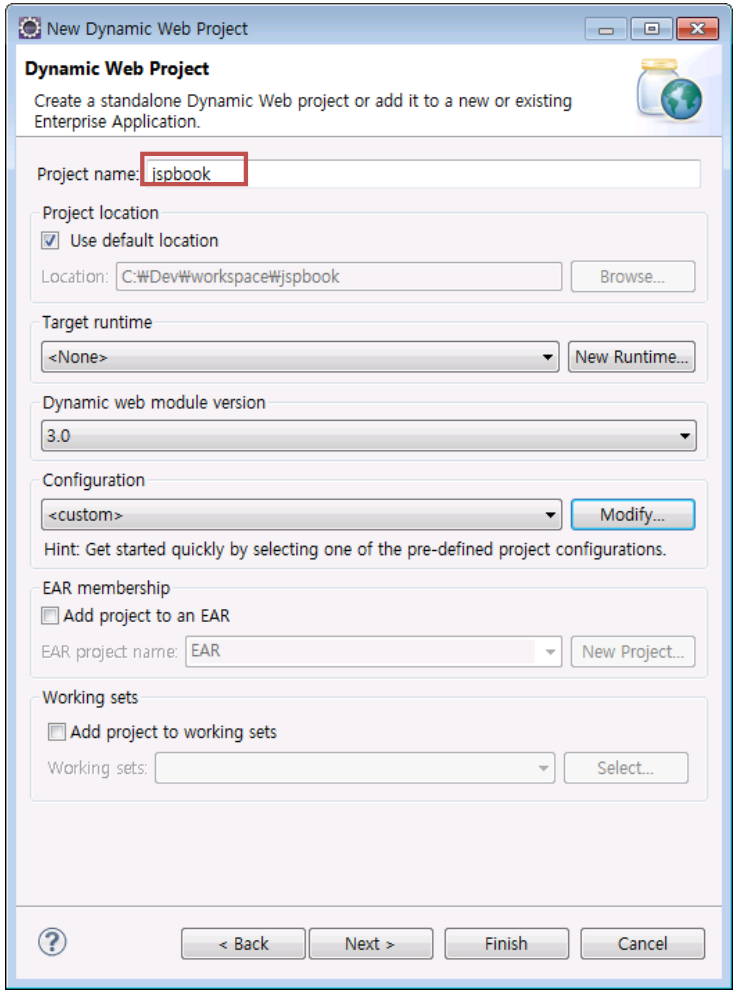
- 이클립스를 실행하고 [File] → [New] → [Project]를 선택하면 미리 정의된 특정 유형의 프로젝트 템플릿을 이용하여 프로젝트를 생성할 수 있다.
- 다이나믹 웹 프로젝트 생성하기
  - **트리 메뉴 중 [Web]을 선택하고 [Dynamic Web Project]를 선택하면 JSP 개발을 위한 프로젝트가 생성된다.**



04. [기본실습]JSP 프로그래밍: Hello World JSP

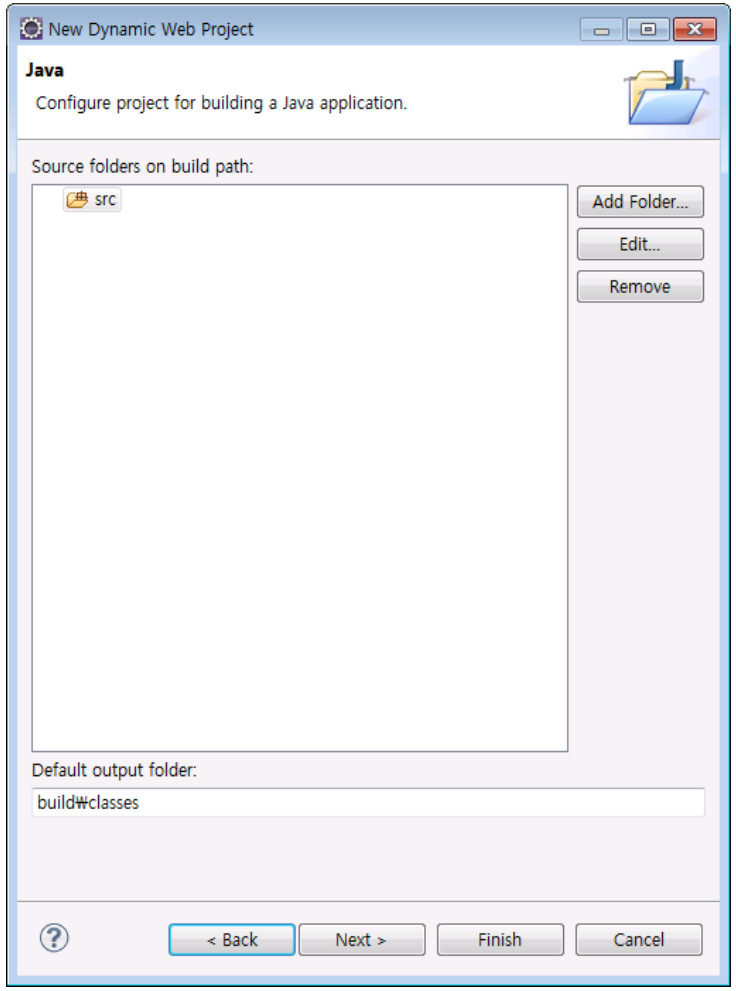
■ 기본 정보 설정하기

- 프로젝트 이름, 웹 모듈 버전 등 프로젝트 기본 정보를 설정한다.



■ 소스 폴더 설정하기

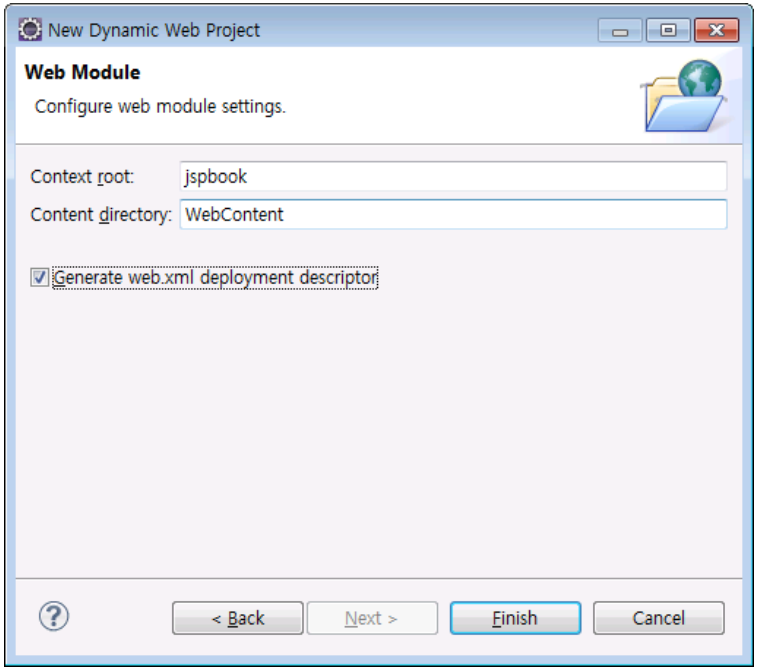
- 자바 클래스 소스 폴더를 지정한다.



04. [기본실습]JSP 프로그래밍: Hello World JSP

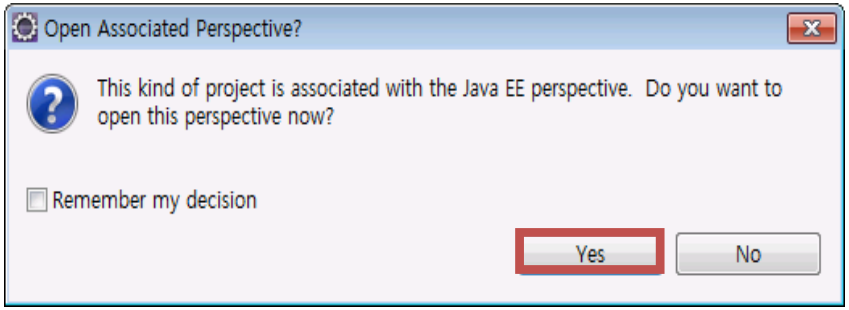
웹 모듈 설정하기

- Context Root
  - 웹 애플리케이션의 메인 접속 경로를 말함.
  - http://localhost:8080/jspbook 과 같이 JSP 실행을 위한 기본 URL에 적용됨.
- Content Directory
  - JSP, HTML, 이미지 등 기본 웹 컨텐츠가 위치하는 디렉터리.
  - 이클립스 프로젝트 구조에서는 [WebContent] 폴더가 기본 값으로 사용됨.

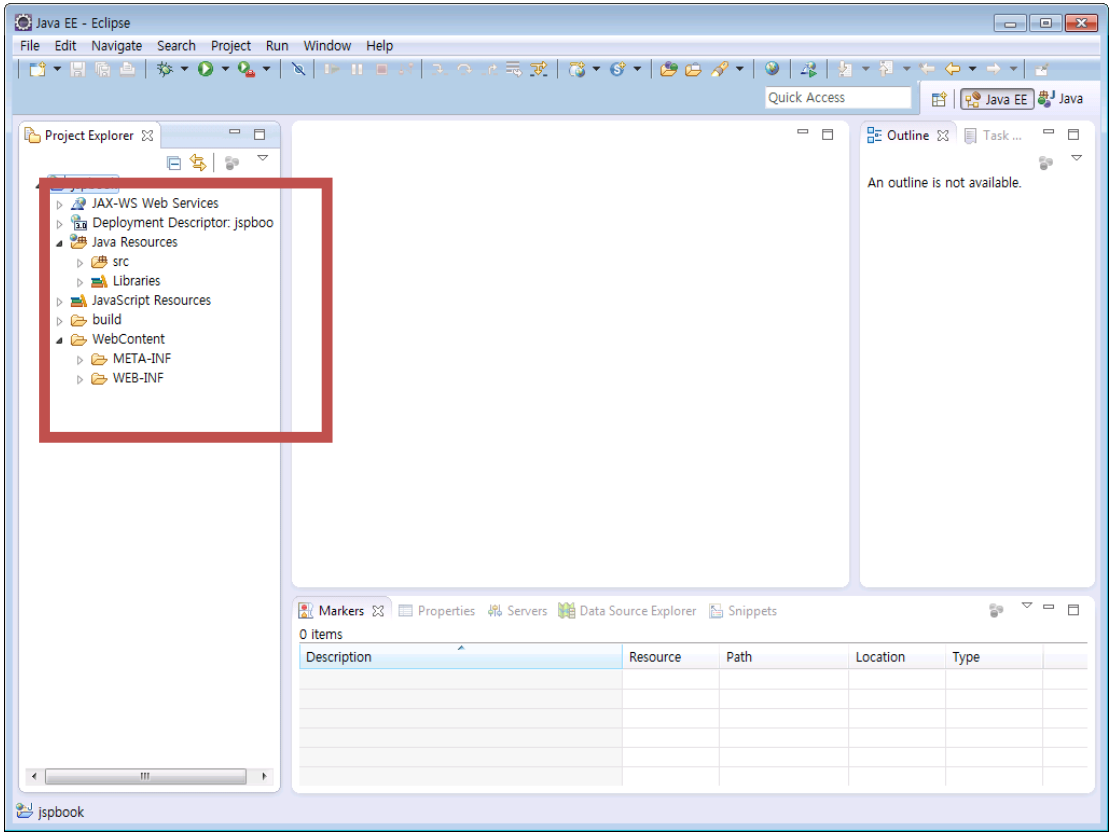


04. [기본실습]JSP 프로그래밍: Hello World JSP

- 퍼스펙티브 변경하기



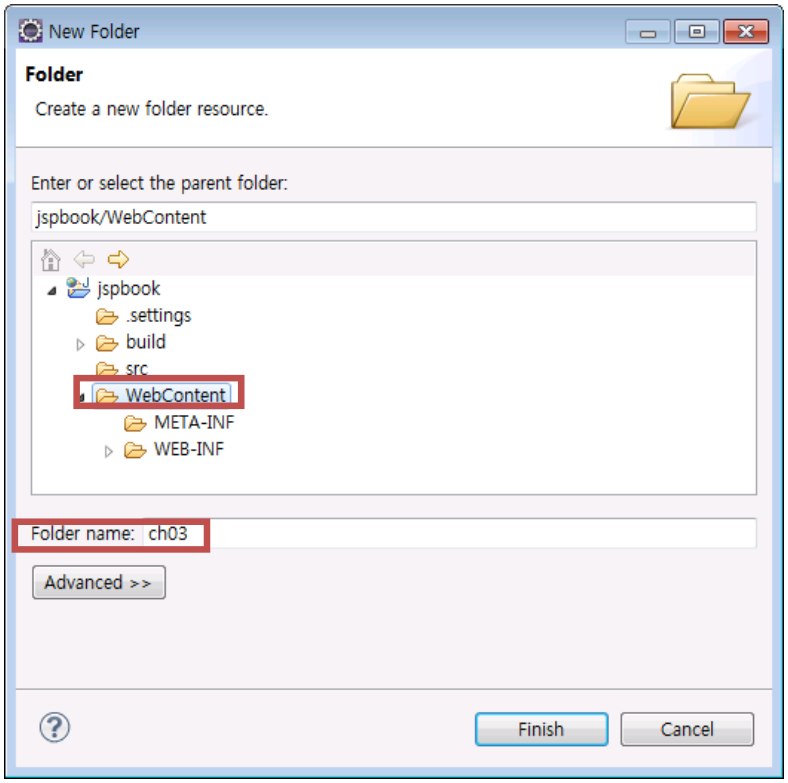
- 생성된 프로젝트 확인



04. [기본실습]JSP 프로그래밍: Hello World JSP

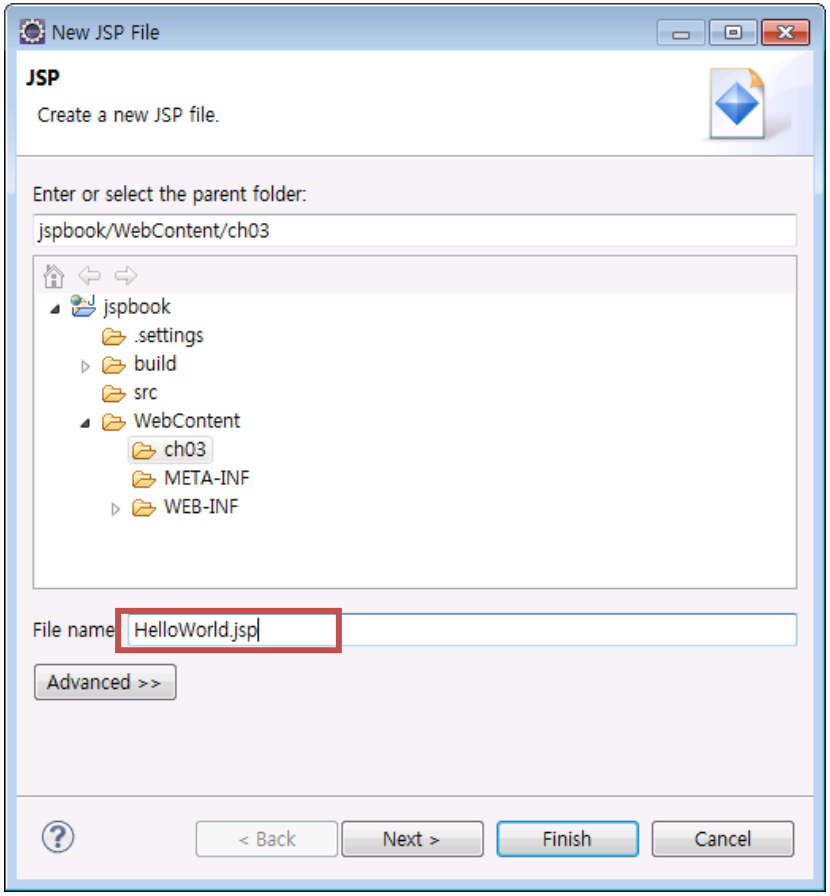
2. Hello World 프로그램 소스 작성

■ 폴더 생성



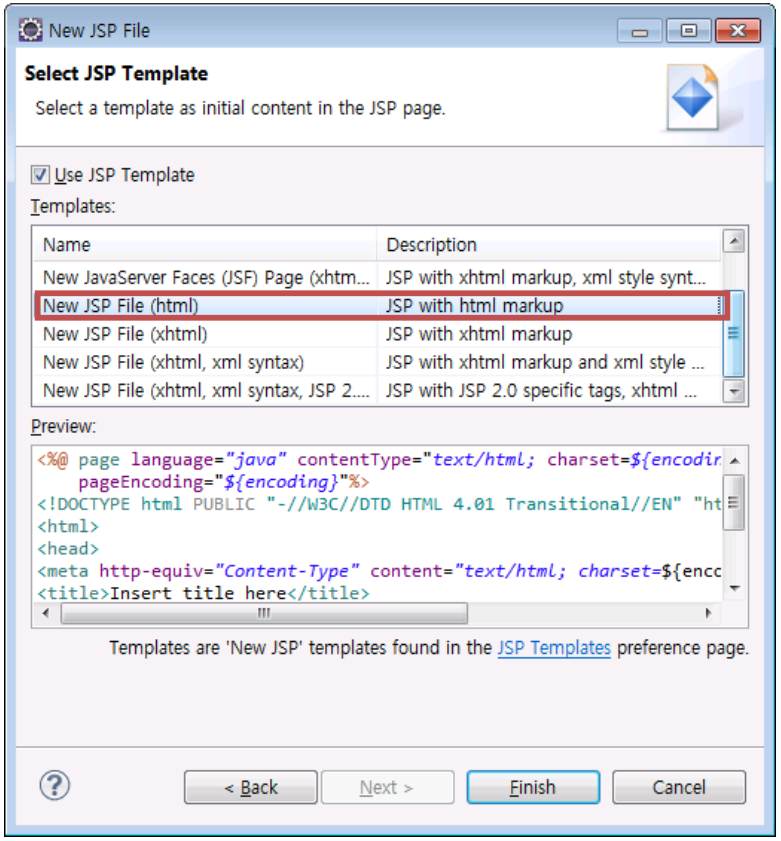
■ JSP 생성

❶ JSP 파일 이름 지정하기



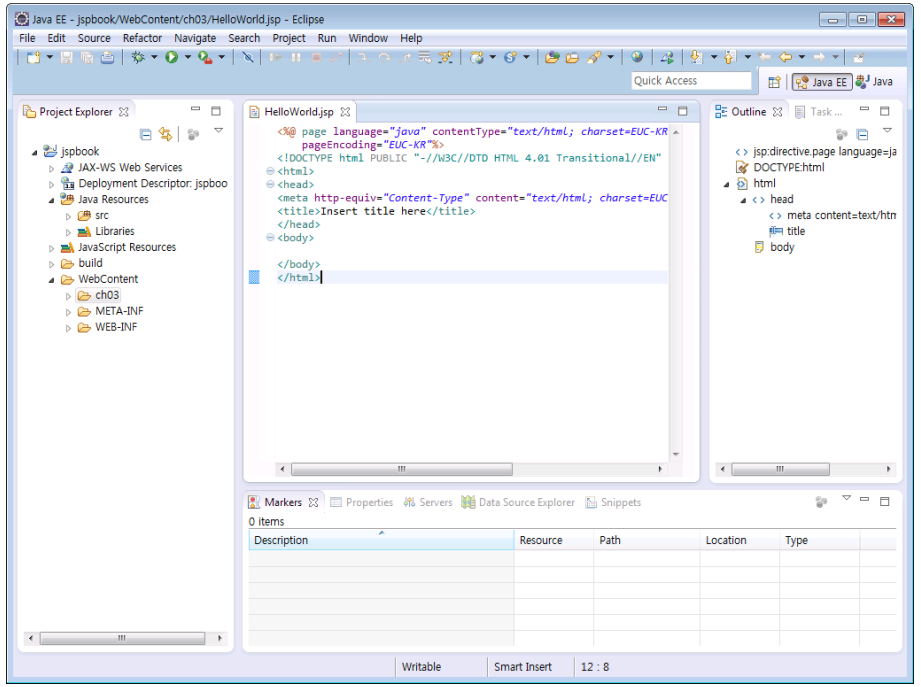
04. [기본실습]JSP 프로그래밍: Hello World JSP

❷ 템플릿 코드 지정하기



[그림 3-14] 템플릿 선택

❸ 생성된 코드 확인하기

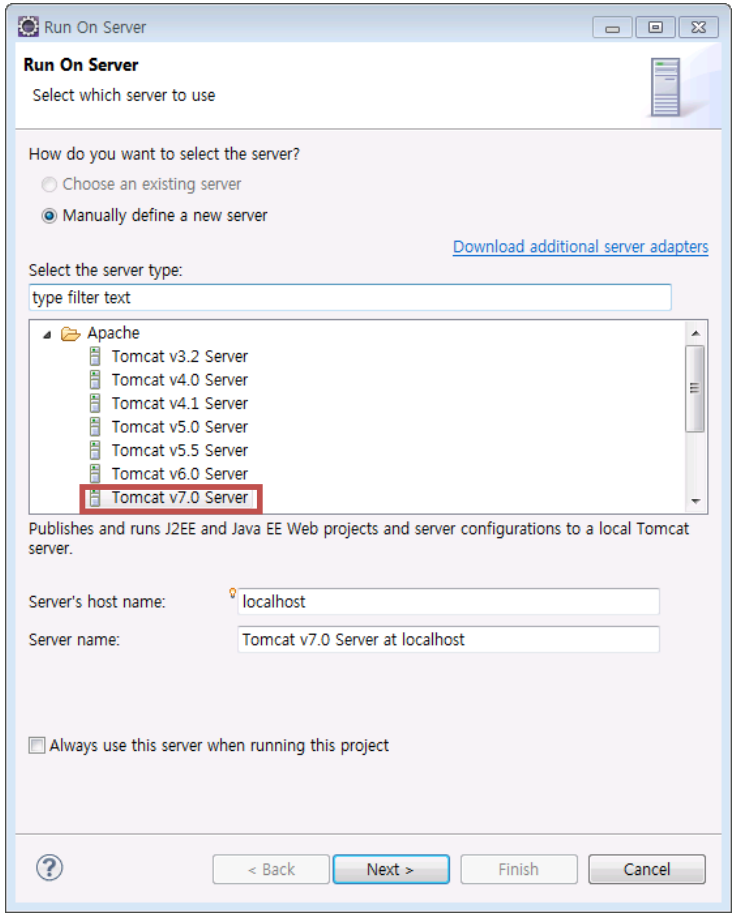


[그림 3-15] 생성된 기본 코드

04. [기본실습]JSP 프로그래밍: Hello World JSP

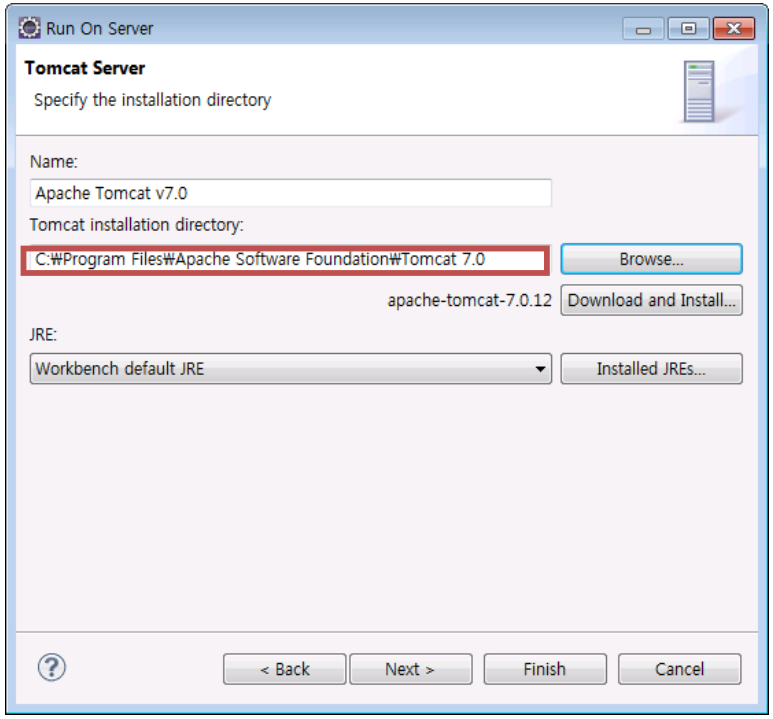
3. 서버 설정 및 실행

■ 서버 설정하기



[그림 3-16] 서버 설정

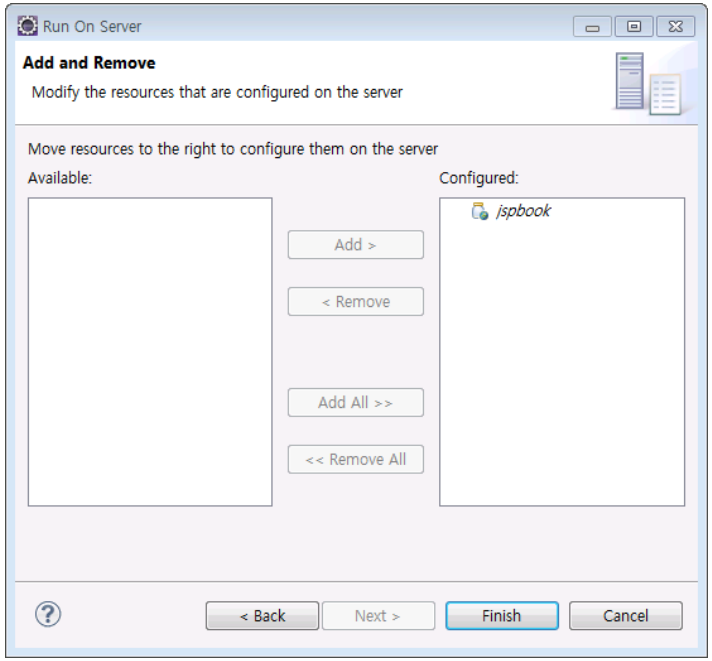
■ 톰캣 폴더 지정하기



[그림 3-17] 톰캣 폴더 지정

04. [기본실습]JSP 프로그래밍: Hello World JSP

■ 실행할 프로젝트 선택하기



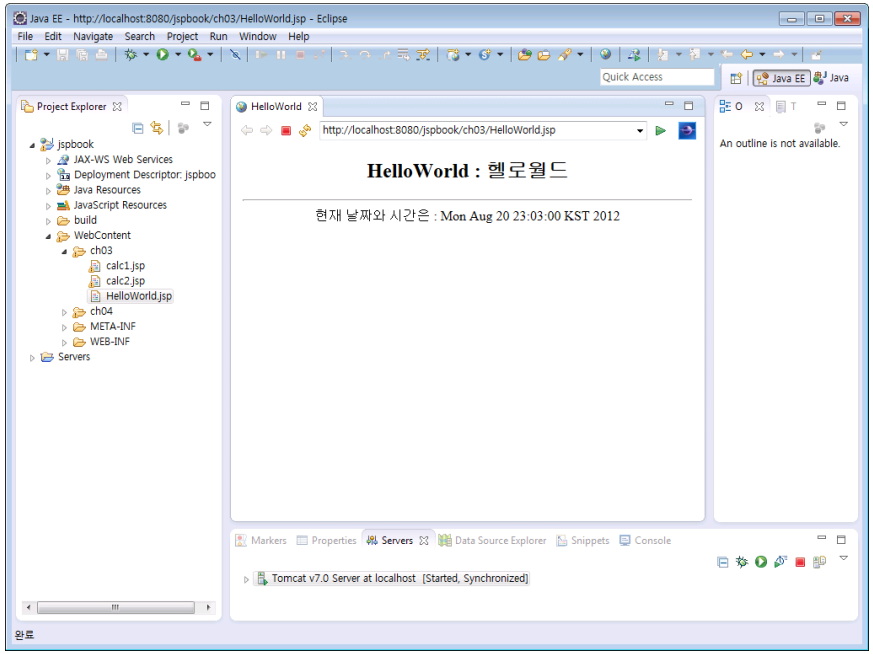
[그림 3-18] 실행 프로젝트 선택

■ 보안 경고 해제하기

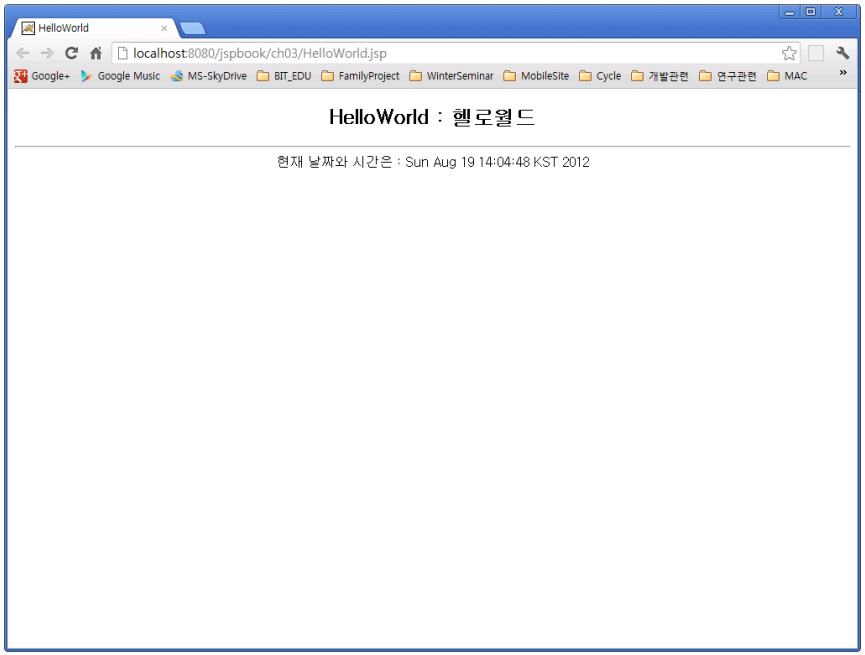
[그림 3-19] 윈도우 보안 경고

04. [기본실습]JSP 프로그래밍: Hello World JSP

실행 결과 확인하기



[그림 3-20] 실행 결과 확인

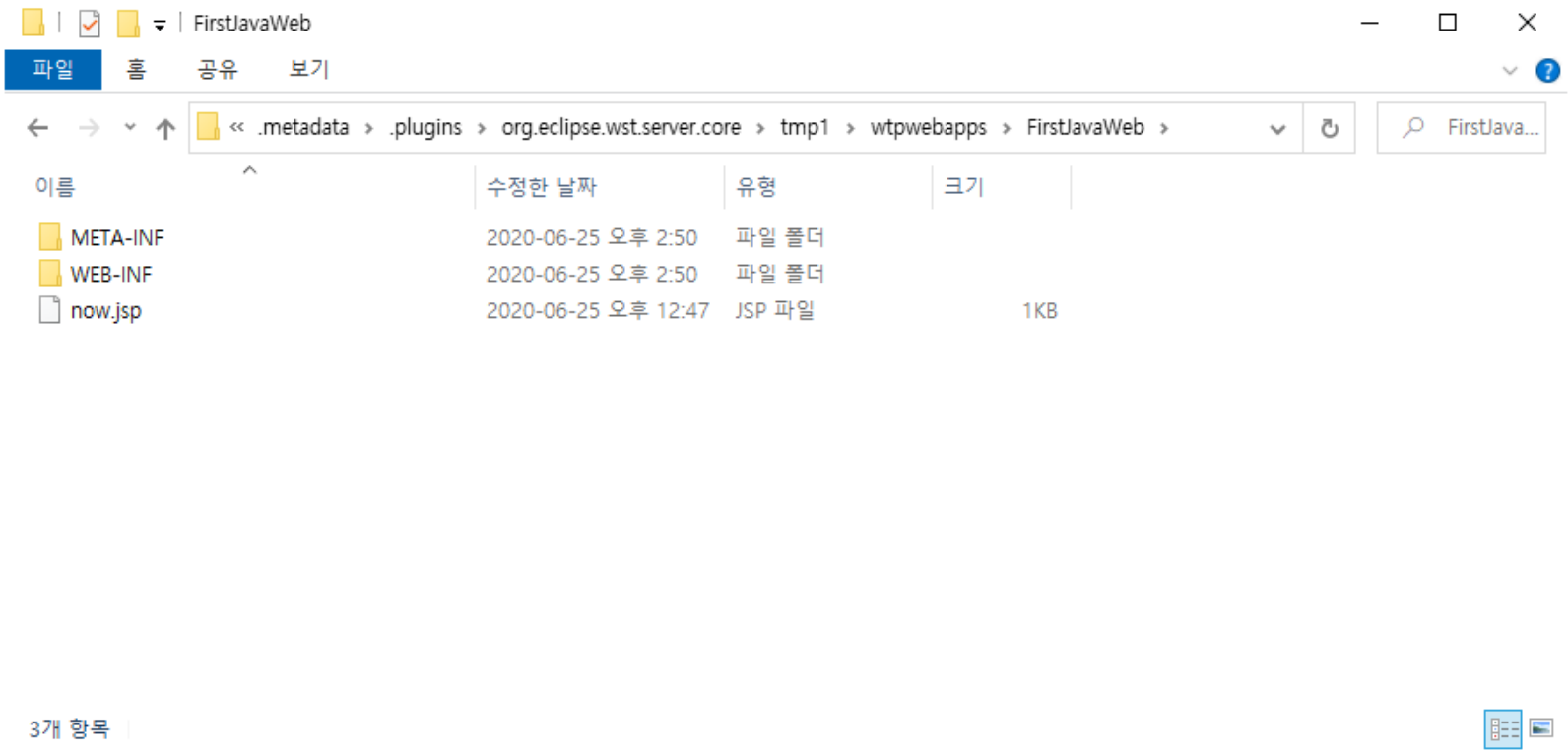


[그림 3-22] 외부 브라우저(크롬)를 이용한 실행 결과

04. [기본실습]JSP 프로그래밍: Hello World JSP

4. 서블릿으로 변환된 소스 확인

- [project workspace ₩.metadata₩.plugins₩org.eclipse.wst.server.core₩tmp1₩wtpwebapps₩FirstJavaWeb] 폴더에 위치



JSP 페이지의 구성 요소

- **디렉티브(Directive)**
- 스크립트: 스크립트릿(Scriptlet), 표현식(Expression),
  선언부(Declaration)
- **표현 언어(Expression Language)**
- **기본 객체(Implicit Object)** request, response, session, application, out
- **정적인 데이터** HTML , text
- 표준 액션 태그(Action Tag)
- 커스텀 태그(Custom Tag)와 **표준 태그 라이브러리(JSTL)**

Tag = > java 문법을 사용하지 않을수있다

EL+JSTL
JSP는 VIEW의 역할

01. 주석

주석 이란 ?

- 주석은 프로그램 소스에 텍스트로 된 간단한 설명문을 넣는 것을 말한다.
- C 언어를 비롯한 대부분의 프로그램 언어가 주석을 사용하기 위한 문법을 제공하고 있다.
- JSP는 특성상 자바, HTML, JSP 코드가 섞여 있으므로 주석도 혼용해서 사용한다.
- HTML 주석 : 클라이언트로 전달되는 주석
  - 일반적인 HTML 문서에서 사용 가능한 주석으로 화면에는 보이지 않지만 브라우저 소스보기를 하면 내용이 노출됨.

```
<!-- 주석입니다. -->
```

- JSP 주석 : 클라이언트로 전달되지 않는 주석
  - JSP 파일에서만 사용 가능한 주석으로 브라우저 소스보기를 해도 내용이 노출되지 않음.

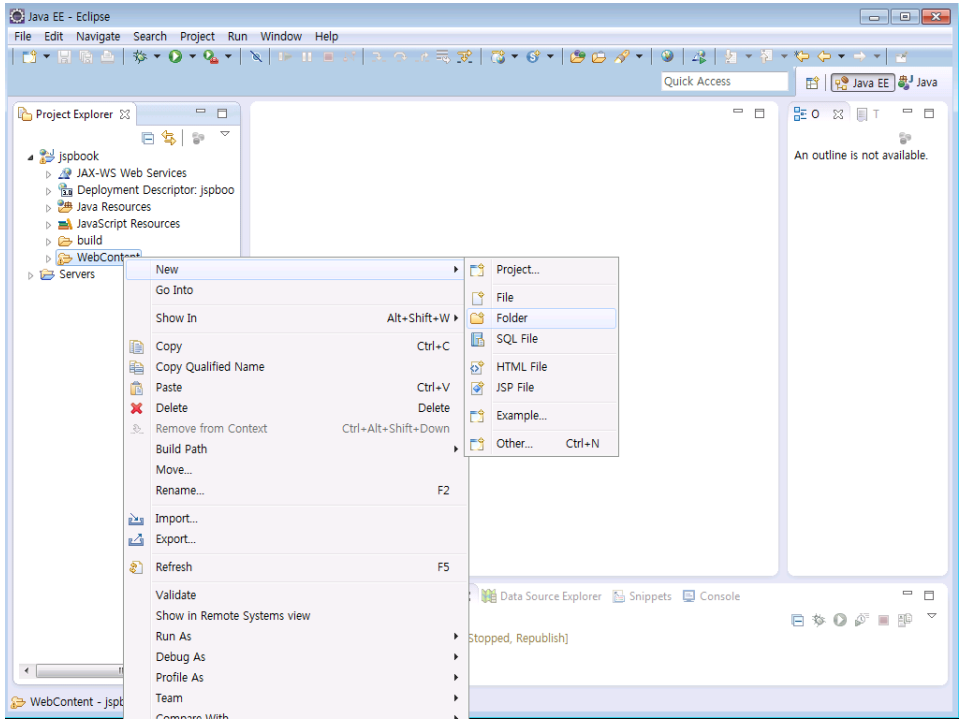
```
<%-- 주석  --%>
```

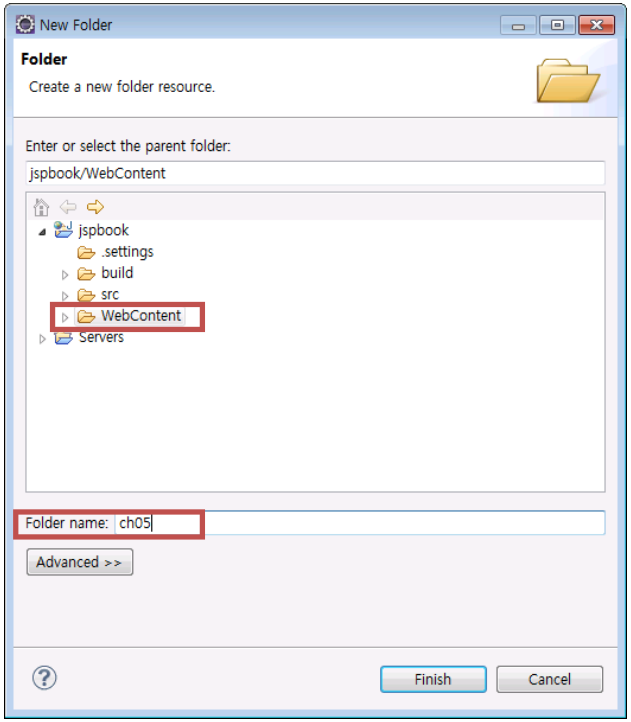
01. 주석

1. 이클립스를 이용한 소스 작성과 실행

■ 이클립스에서 작업 준비하기

❶ [WebContent] → [ex] 폴더를 만든다.





02. 지시어

지시어(Directives)란?

- 지시어(Directives)는 JSP 파일의 속성을 기술하는 JSP 문법.
- JSP 컨테이너에게 해당 페이지를 어떻게 처리해야 하는지 전달하기 위한 내용을 담고 있다.
- 지시어는 크게 page, include, taglib으로 나눌 수 있으며, 각각에서 다루는 속성이 다르다.

include를 이용하면
화면을 분할 시킬수있다.

1. page 지시어

- page 지시어는 현재 JSP 페이지를 컨테이너에서 처리하는 데 필요한 각종 속성을 기술하는 부분.
- 보통 JSP 페이지 맨 앞에 위치함.

```
<%@ page 속성1="속성값1" 속성2="속성값2“ ... %>
```

- 여러 줄에 나누어 작성할 수도 있음.

```
<%@ page contentType="text/html;charset=UTF-8"
    import="javax.sql.*, java.util.*" errorPage="error.jsp"%>
<%@ page import="java.util.*" %>
```

02. 지시어

- 이클립스 개발 도구를 이용해 jsp 파일을 생성하는 경우 기본적인 page 지시어는 자동 생성됨.

속성	설명	기본 설정 값
language	스크립트 언어의 유형을 정한다.	java
import	JSP 내에서 사용할 외부 자바 패키지나 클래스의 불러오기(import)를 정한다.	–
session	세션의 사용 유무를 정한다.	true
buffer	버퍼의 크기를 정한다.	8KB
autoFlush	버퍼의 내용을 자동으로 비운다.	true
isThreadSafe	단일 스레드 모델을 사용함으로써 동시성 제어 여부를 정한다.	true
info	JSP 페이지에 대한 설명이다.	–
errorPage	현재 페이지에서 오류가 발생할 경우 호출될 페이지를 지정한다.	–
isErrorPage	오류만을 처리하는 페이지로 지정한다.	false
contentType	MIME 형식 지정 및 캐릭터셋을 설정한다.	text/html;charset=ISO-8859-1
pageEncoding	contentType과 동일한 기능을 한다.	ISO-8859-1

HttpSession

utf - 8

utf - 8

02. 지시어

Page 지시어와 JSP의 한글 처리

- page 지시어에서 중요한 부분 중 하나는 한글 처리 부분임.
- JSP 에서는 다음과 같이 3단계로 캐릿터셋을 설정함.

우선순위	설정 방법	비고
1	server.xml 파일에서 정의한다.	Connector 설정에 URI 요청에 대한 캐릭터셋을 지정할 수 있다. 작성한 소스를 다른 서버에서 실행할 경우, 해당 서버 설정에 따라 한글이 깨질 수도 있다.
2	각 JSP 파일, page 지시어의 pageEncoding 속성에서 정의한다.	페이지 단위의 설정이므로 애플리케이션 설정과 무관하게 한글을 처리할 수 있다. 단, 서버가 JSP 2.0 스펙을 지원하지 않는 예전 버전이라면 pageEncoding 속성을 인식하지 못해 한글이 깨질 수 있다.
3	각 JSP 파일, page 지시어의 contentType 속성에서 정의한다.	클라이언트에 대한 응답 결과를 지정하는 부분으로 브라우저에게 전달할 캐릭터셋을 지정할 수 있다.

- 위 설정에서 캐릭터셋 설정을 찾지 못할 경우 ISO8859-1을 적용한다.
- 페이지 지시어에 다음과 같이 한글 속성 설정
  - pageEncoding="UTF-8", contentType="text/html;chatsert=UTF-8"

02. 지시어

import

- import 는 JSP 스크립트 부분에서 자바 클래스를 사용하는 경우 해당 클래스의 패키지에 대한 import 설정으로 기본적으로 자바에서와 동일하다.
- 다만 패키지 import 구분을”,” 을 이용하거나 라인 단위로 작성해야 한다.
- 다음은 page 지시어에서 import 속성의 다양한 사용 예이다.

```
<%@ page import="java.sql.*,java.util.*" %>
```

```
<%@ page import="java.sql.*" %>
<%@ page import="java.util.*" %>
```

```
<%@ page
        import="java.sql.*,
        java.util.*"
%>
```

02. 지시어

(일정시간이 지나면소멸된다.)

session 사용자 처음서버 접속

- 세션은 웹 브라우저와 웹 서버가 지속적인 클라이언트 인식을 위해 필요한 정보를 임시로 저장해두는 방법
- 주로 웹 사이트에 로그인하거나 쇼핑몰에서 장바구니 등을 구현할 때 사용된다.
- 기본 값이 true(세션을 사용한다)이므로, 일부러 사용을 제한할 목적이 아니라면 별도로 설정하지 않아도 됨. 특정 목적이 있으면 false로 정의
- 세션과 관련한 자세한 내용은 뒤 다른 챕터에서 다룸.

```
<%@ page session="true" %>
```

buffer

- JSP 페이지 데이터를 출력하기 위한 JspWriter 즉 out 내장객체의 버퍼 크기를 지정.
- 기본값은 8KB 이고 JSP 페이지에 동적으로 많은 내용이 포함될 경우 버퍼 크기 조정이 필요할 수도 있으나 일반적으로는 변경하지 않아도 됨.

02. 지시어

autoFlush

- autoFlush는 버퍼를 자동으로 비울 것인지를 지정하는 속성으로, 기본 값은 true다.
- 버퍼 속성에 지정되어 있는 크기만큼 버퍼를 유지하고 있다가 버퍼가 다 차면 자동으로 전송한다.

```
<%@ page autoFlush="true" %>
```

isThreadSafe

- 기본적으로 서블릿은 스레드로 동작하기 때문에 스레드로 인한 동기화 문제를 해결하기 위한 옵션임.
- 기본값은 true로, 일반적으로 false로 설정하는 경우는 거의 없다.

```
<%@ page ifThreadSafe="true" %>
```

info

- 해당 JSP에 대한 간단한 설명으로 저작권이나 작성일 등 간단한 정보 기술에 사용.

```
<%@ page info="JSP Example" %>
```

02. 지시어

errorPage, isErrorPage

- 두 속성은 jsp파일의 오류 처리를 위한 것으로, errorPage는 현재 페이지에 오류 발생시 호출할 페이지를 지정하는 속성이고 isErrorPage는 오류 처리를 위한 전용 페이지임을 알리는 속성이다.

- errorPage지정을 통해 보다 효과적으로 페이지 오류를 관리할 수 있다.
  - errorPage : 일반적인 JSP 파일에 사용

```
<%@ page errorPage="오류_처리_파일.jsp" %>
```

  - isErrorPage : 오류 처리 파일에만 사용

```
<%@ page isErrorPage="true" %>
```

02. 지시어

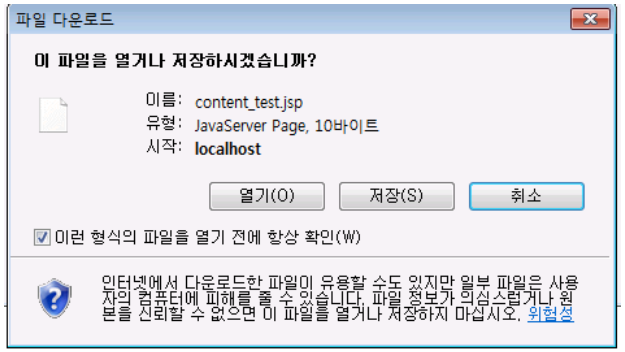
contentType

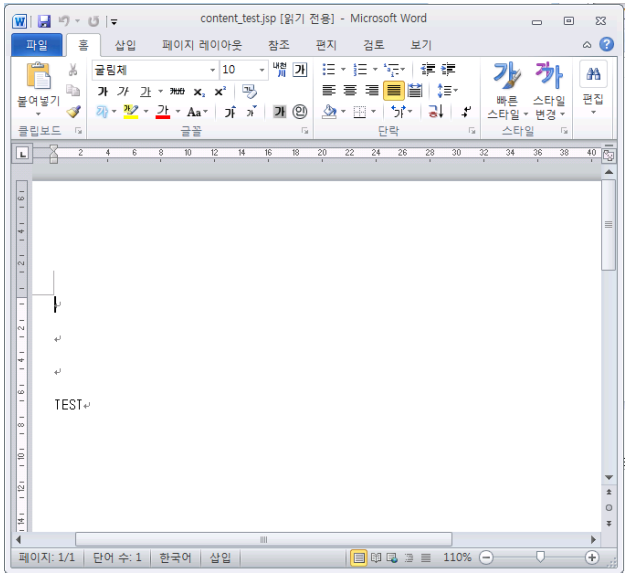
- 현재 JSP 페이지를 클라이언트에서 처리하기 위한 콘텐츠 유형을 지정하는 부분.
- 윈도우에서 파일 확장자(.doc, .hwp등)에 따라 연결 프로그램이 동작하는 것과 마찬가지로 웹 브라우저에서도 contentType에 따라 전달되는 내용을 어떻게 처리할지 결정할 수 있다.

```
<%@ page contentType="text/html" %>
```

- text/html 이 아니라 application/msword 로 지정할 경우 브라우저는 서버가 전달하는 콘텐츠를 ms word 문서로 인식해 처리할 것은 사용자에게 요청함.

```
<%@ page contentType="application/msword" %>
```





02. 지시어

pageEncoding

- pageEncoding은 컨테이너에서 처리할 JSP 파일의 인코딩을 설정.

```
<%@ page pageEncoding="UTF-8" %>
```

- JSP 2.0 스펙에 추가된 속성으로, 이전 버전을 지원하는 컨테이너의 경우에는 사용할 수 없다.

02. 지시어

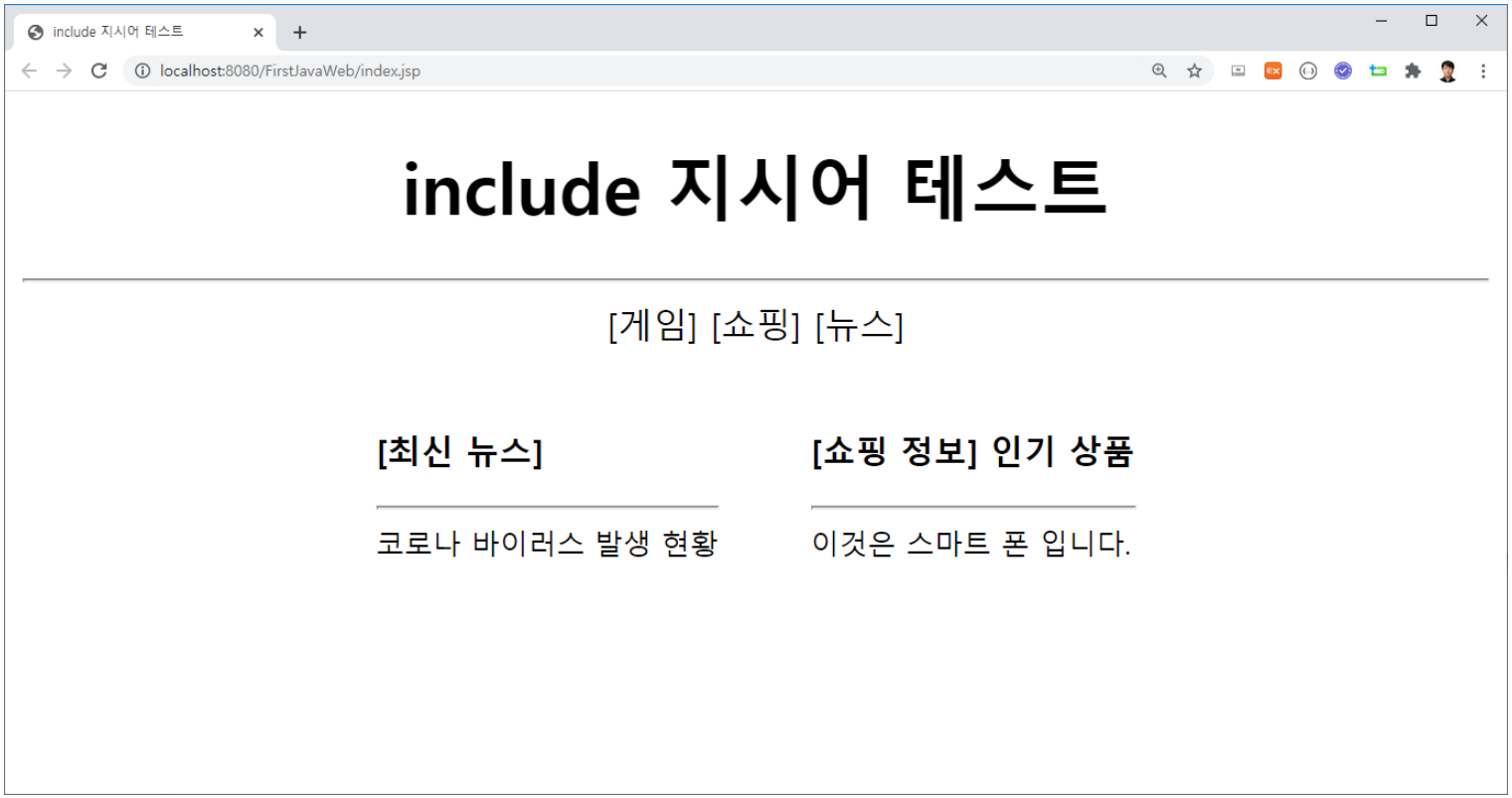
2. include 지시어

- include 지시어는 현재 JSP 파일에 다른 HTML이나 JSP 문서를 포함하기 위한 기능을 제공.
- include 지시어는 다음 절에서 살펴볼 include 액션과 비슷한 기능을 한다.

```
<%@ include file="포함할 파일_이름" %>
```

- 네이버와 같은 인터넷 포털사이트의 화면처럼 여러 정보의 조합으로 한 화면을 구성할 때 유용하게 사용됨.
- include 지시어를 사용하면 기능 혹은 화면을 모듈화할 수 있어 화면 구성이나 재활용이 용이하다.

02. 지시어


include 지시어 테스트
localhost:8080/FirstJavaWeb/index.jsp
include 지시어 테스트
[게임] [쇼핑] [뉴스]
[최신 뉴스]
코로나 바이러스 발생 현황
[쇼핑 정보] 인기 상품
이것은 스마트 폰 입니다.

02. 지시어

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8" errorPage="error.jsp"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<TITLE>include 지시어 테스트</TITLE>
</HEAD>
<BODY>
<div align="CENTER">
<H2>include 지시어 테스트</H2>
<HR>

<%@ include file="menu.jsp"%>
<P>
<table border=0 cellpadding=5 cellspacing=1>
<tr>
<td><font size=-1><%@include file="news.jsp"%></font></td>
<td width="20"> </td>
<td><font size=-1><%@include file="shopping.jsp"%></font></td>
</tr>
</table>

</div>
</BODY>
</HTML>
```

03. 액션

mvc 등장후 많이 사용안함
spring + mvc 를 사용하면 굳이 액션태그를 사용할필요없다

1. JSP 액션의 종류

- JSP 액션은 JSP 고유 기능으로 빈즈 클래스 연동 및 동적 페이지 관리를 위한 기능을 제공함.
- <jsp:action_name attribute="value" /> 형태를 가짐.
- 주로 사용하는 액션은 useBean, get/setProperty 이며 자바 클래스와의 연동을 위해 사용함.
- 액션(Action)은 JSP 주요 구성요소 중 하나로 다음과 같은 기능을 지원한다.
  - JSP 페이지간 흐름 제어
  - 자바 애플릿 지원
  - 자바 빈즈 컴포넌트와 JSP 상호작용 지원
- 특히 useBean 액션은 JSP에서 자바 빈즈 클래스와의 연동을 지원해주는 액션으로 잘 알아 둘 필요가 있다.
- include 액션은 단순히 페이지를 포함하는 것 뿐만 아니라 파라미터를 포함될 페이지로 전달하는 것이 가능함.
  - 사용 예) <jsp:param name="user" value="홍길동" />

03. 액션

(실행의 결과를 가져온다)차이점

액션	사용 예	기능
include	<jsp:include page="xx.jsp" />	다른 페이지를 현재 페이지에 포함시킨다.
forward	<jsp:forward page="xx.jsp" />	현재 페이지의 제어를 다른 페이지로 전달한다.
useBean	<jsp:useBean scope="page" id="cls" class="xx.MyBean" />	xx패키지의 MyBean 클래스를 cls라는 이름으로 page 범위에서 사용할 것을 선언한다.
setProperty	<jsp:setProperty name="cls" property="xxx" />	useBean으로 선언된 빈즈 클래스의 setxxx( ) 메서드를 호출한다.

객체생성
데이터를 공유할때 사용

액션	사용 예	기능
getProperty	<jsp:getProperty name="cls" property="xxx" />	useBean으로 선언된 빈즈 클래스의 getxxx( ) 메서드를 호출한다.
plugin	<jsp:plugin type="applet/bean" code="class"> </jsp:plugin>	애플릿이나 빈즈 클래스를 플러그인 형태로 로딩한다.
param	<jsp:param name="user" value="홍길동" />	include, forward 액션에서 사용할 수 있는 파라미터를 설정한다.

03. 액션

2. include 액션

- include 액션은 다른 파일을 불러온다는 측면에서 include 지시어와 개념이 유사.
- include 지시어는 해당 파일을 포함시킨 후 컴파일하는 것에 비해, include 액션은 실행 시점에서 해당 파일을 호출하여 그 결과를 포함한다는 점에서 차이가 있음.
- 동적으로 파일들을 핸들링 하기 때문에 과도한 사용은 성능상에 문제를 줄 수 있음.
- include 액션은 동적인 페이지 를 포함시킬 경우에 사용하는 것이 좋고, include 지시어는 잘 바뀌지 않는 정적인 페이지를 포함 할 때 사용하는 것이 좋다.

```
<jsp:include page="포함할 파일_이름" />
```

- 사용 예

```
<jsp:include page="footer.jsp">
    <jsp:param name="email" value="test@test.net" />
    <jsp:param name="tel" value="000-000-0000" />
</jsp:include>
```

03. 액션


footer.jsp
include_action.jsp
include_action.jsp
http://localhost:8080/jspbook/ch05/include_action.jsp
include_action.jsp 에서footer.jsp 호출
include_action.jsp 에서 출력한 메시지 입니다.
footer.jsp 에서 출력한 메시지 입니다.
test@test.net, 000-000-0000 000-000-0000

03. 액션

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<TITLE>include action 테스트</TITLE>
</HEAD>
<BODY>
<H2>include_action.jsp 에서footer.jsp 호출</H2>
<HR>
include_action.jsp 에서 출력한 메시지 입니다.
<BR>

<jsp:include page="footer.jsp">
<jsp:param name="email" value="test@test.net" />
<jsp:param name="tel" value="000-000-0000" />
</jsp:include>
</BODY>
</HTML>
```

03. 액션

3. forward 액션

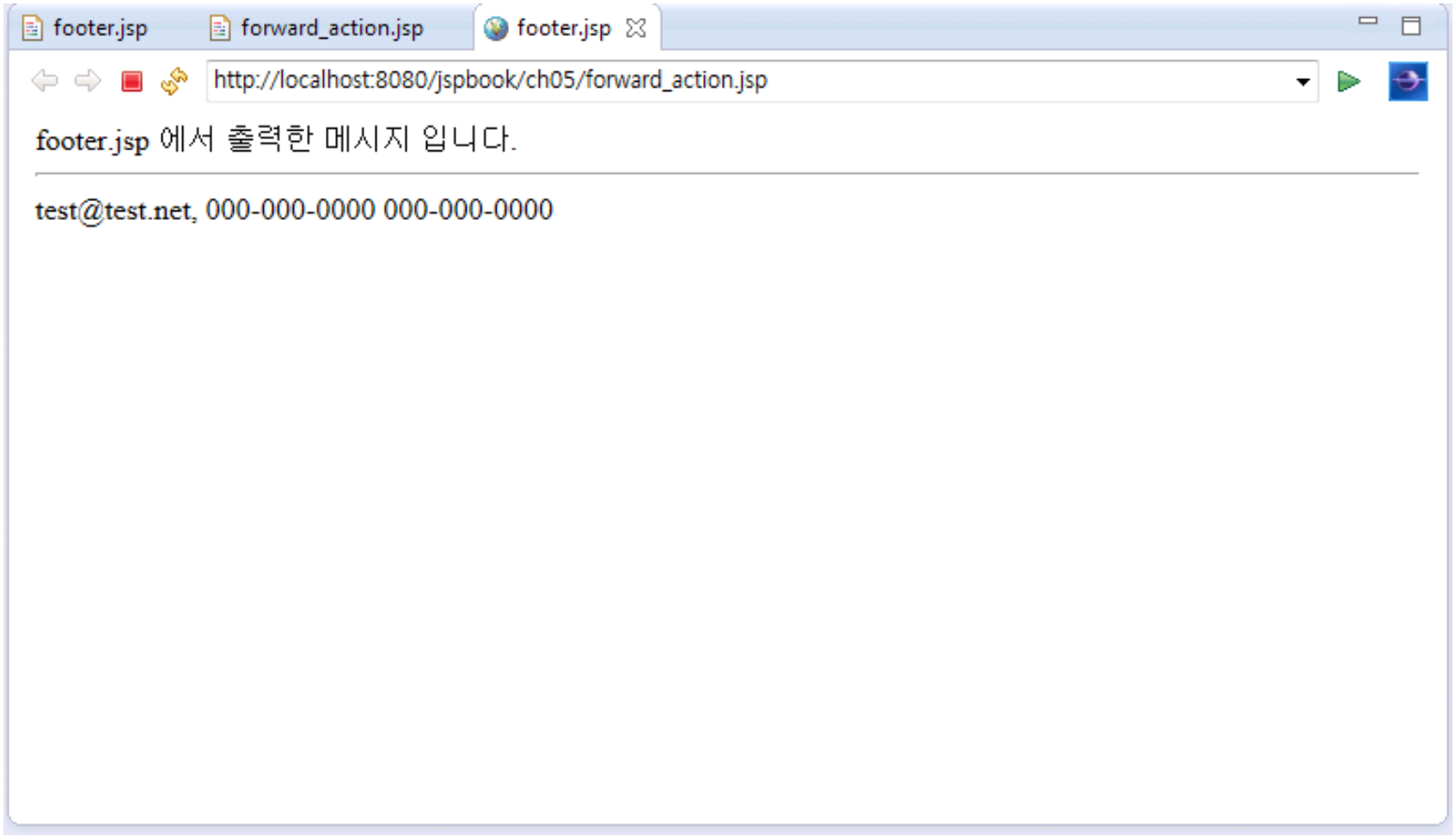
- forward 액션은 include 액션과 사용법은 유사하지만 요청 페이지를 다른 페이지로 전환할 때 사용한다.
- response 내장객체의 sendRedirect()와 유사 하지만 포워드된 페이지에 파라미터를 전달할 수 있다는 점에서 차이가 있다.
- 브라우저 URL 창에는 최초 요청 페이지가 표시 되기 때문에 처리 페이지 정보를 숨기거나 MVC 패턴의 컨트롤러와 같이 특정 기능 수행 후 다른 페이지로 이동해야 하는 경우 유용하게 사용할 수 있다.

```
<jsp:forward page="포워딩할 파일_이름" />
```

- 사용 예)

```
<jsp:forward page="footer.jsp" />
<jsp:param name="email" value="test@test.net" />
<jsp:param name="tel" value="000-000-0000" />
</jsp:forward>
```

03. 액션


footer.jsp
forward_action.jsp
footer.jsp
http://localhost:8080/jspbook/ch05/forward_action.jsp
footer.jsp 에서 출력한 메시지 입니다.
test@test.net, 000-000-0000 000-000-0000

03. 액션

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<TITLE>ch05 : forwared action 테스트</TITLE>
</HEAD>
<BODY>
<H2>index3.jsp 에서 footer.jsp 호출</H2>
<HR>
index3.jsp 의 모든 내용은 출력되지 않습니다.
<% 10;%>
<jsp:forward page="footer.jsp">
<jsp:param name="email" value="test@test.net" />
<jsp:param name="tel" value="000-000-0000" />
</jsp:forward>
</BODY>
</HTML>
```

03. 액션

4. useBean 액션

- 액션에서 가장 중요한 부분으로, JSP 빈즈를 다루는 7장에서 자세히 살펴볼 것이다.
- 여기서는 기본 표기 방법만을 살펴보기로 한다.
- 사용법

```
<jsp:useBean id="변수_이름" class="빈즈 클래스_이름"/>
<jsp:setProperty name="변수_이름" property="속성_이름"/>
<jsp:getProperty name="변수_이름" property="속성_이름"/>
```

- useBean 액션은 빈즈 클래스를 사용하기 위한 구문이며 class 에 지정된 자바 빈즈 클래스를 id 라는 이름으로 사용할 수 있도록 해준다.
- get/setProperty 액션은 브라우저에서 빈즈 클래스의 멤버 변수로 값을 저장하거나 가져오기 위한 구문 이다.
- get/setProperty는 빈즈 클래스의 getter/setter 메서드와 연동된다.

04. 선언과 표현식

1. 선언

- JSP 페이지에서 메서드나 멤버변수를 선언하기 위한 구문.
- JSP 가 서블릿으로 변환된 자바 코드에서는 모든 내용이 _jspService() 메서드에 들어가기 때문에 JSP 에서 선언한 변수는 로컬변수가 되고 메서드 안에서 다른 메서드를 선언하는 자바 문법상 잘못된 것이므로 컴파일 에러가 발생하게 됨.
- <%! %>는 JSP 페이지에서 이러한 제약 사항 없이 멤버변수와 메서드 선언을 가능하게 함.
- 구조적으로 JSP 에서 자바 코드를 사용하는 것은 권장되지 않기 때문에 선언문의 사용 역시 권장되지 않음.
- 사용 예)

```
<%!
    // 멤버변수 선언이나 메서드 선언이 올 수 있다.
    String str = "test";
    public boolean check() {
    return false;
    }
%>
```

04. 선언과 표현식

2. 표현식

- 표현식(Expression)은 이미 여러 소스를 통해 많이 살펴본 것처럼 <%= %>를 사용해서 간단한 데이터 출력이나 메서드 호출 등에 이용한다.
- 코드 마지막에 ';(세미콜론)'을 사용하지 않는다는 것에 주의하도록 한다.
- 표현식은 결국 out.println() 으로 변환되기 때문에 out.println()의 인자로서 적합한 형태로 사용해야 함.
- 메서드 호출, 변수출력, 사칙연산 및 문자열 결합 등이 가능하다.
- 표현식 보다는 표현언어(Expression Lanugage) 사용을 권장한다. → 10장
  EL
- 사용 예)

```
메서드 호출 : <%= calculator() %>
변수 출력 : <%= result %>
사칙 연산과 문자열 결합 : <%= "i+2="+(i+2)+"입니다" %>
```

05. 스크립트릿

1. 스크립트릿이란?

- 스크립트릿(Scriptlet) 은 JSP 문서 내에서 자바 코드를 기술할 수 있는 부분으로 JSP의 가장 큰 특징 중 하나.
- 초기 JSP 발전에 큰 기여를 하였으나 지금은 JSP내에서 스크립트릿 사용은 권장되지 않음.
- MVC 패턴에 따라 웹 프로그램을 개발하게 되면 JSP는 View의 역할을 하게 되고 표현언어, JSTL, 커스텀 태그 라이브러리, JSP 빈즈가 주로 사용된다.
- UI(AWT, Swing 등)을 제외한 거의모든 자바 클래스 라이브러리를 사용한 프로그래밍이 가능함.

```
<% // 로컬 변수 선언이나 프로그램 로직이 올 수 있다.
    String str = "test";
    for (int i=0; i < 10; i++)
            out.println(i);
%>
```